12ème École Thématique de Simulation Numérique

« La simulation numérique à l'ère de l'IA générative »



28 septembre – 2 octobre 2026

Maison d'hôtes de Cadarache

Organisation :

LRC MESO
(Centre Borelli, ENS Paris-Saclay et CEA-DAM Île-de-France)

Présentation

L'intelligence artificielle générative transforme en profondeur les sciences de l'ingénieur, la recherche et les pratiques de développement logiciel. Cette école d'été a pour ambition de fournir aux participants une compréhension à la fois théorique et pratique des interactions entre IA générative et simulation numérique.

Au cours de la semaine, nous reviendrons sur les grandes étapes de l'évolution récente de l'IA, depuis le renouveau des réseaux de neurones profonds au début des années 2010 jusqu'à l'émergence des grands modèles de langage (LLMs) qui révolutionnent aujourd'hui de nombreux domaines d'application. Les principes fondamentaux de ces approches, leurs capacités, mais aussi leurs limites seront présentés.

Une attention particulière sera portée à l'utilisation des LLMs dans le cadre de la programmation scientifique. Les participants découvriront comment ces outils peuvent assister le développement, la mise en place d'une documentation, la génération de tests unitaires, le débogage et l'optimisation de codes de calcul, tout en conservant l'esprit critique indispensable à leur utilisation.

L'école privilégiera une approche concrète grâce à plusieurs séances de travaux pratiques. Ces ateliers permettront de mettre en œuvre des techniques d'IA dans des codes de simulation numérique et d'explorer des cas d'usage représentatifs des problématiques rencontrées dans l'industrie et la recherche.

Enfin, parce que les avancées technologiques s'accompagnent de responsabilités nouvelles, une réflexion sera menée sur les questions éthiques associées à l'utilisation de l'intelligence artificielle : fiabilité des résultats, biais, propriété intellectuelle, confidentialité des données et impact sur les pratiques professionnelles.

Conférenciers

Matthieu Ancellin (Mews Labs)
Pierre Andraud (LRC MESO, CEA ENS Paris-Saclay)
Daniel Chauveheid (Mews Labs)
Florian De Vuyst (UTC)
Inna Kucher (CEA Saclay)
David Lugato (CEA CESTA)
Valentina Mejia Gallon (Mews Labs)
Payel Mukhopadhyay (University of Cambridge)
Grégory Robert (CEA Gramat)
Thierry Valet (MPhysX OÜ)

Programme

	28/09/2026	29/09/2026	30/09/2026	01/10/2026	02/10/2026
09H00 - 10H30	ACCUEIL DES PARTICI-PANTS	Matthieu Ancellin COURS	Thierry Valet COURS	David Lugato COURS	Daniel Chau-veheid TP (1)
		PAUSE	**PAUSE**	**PAUSE**	**PAUSE**
11H00 - 12H30		Pierre Andraud COURS	Valentina Mejia Gallon COURS	Payel Mukhopadhyay COURS	Daniel Chauveheid TP (2)
	DEJEUNER	**DEJEUNER**	**DEJEUNER**	**DEJEUNER**	**DEJEUNER**
14H00 - 15H30	Inna Kucher CONFERENCE INAUGURALE (1)	Florian De Vuyst COURS (1)	Florian De Vuyst TP (1)	Payel Mukhopadhyay TP	
	PAUSE	**PAUSE**	**PAUSE**	**PAUSE**	
16H00 - 17H30	Inna Kucher CONFERENCE INAUGURALE (2)	Florian De Vuyst COURS (2)	Florian De Vuyst TP (2)	Grégory Robert COURS + DEMO	

L'intelligence artificielle : comment en est-on arrivé là ?

Inna Kucher (CEA Saclay)

Cette introduction se concentre sur l'évolution des méthodes d'intelligence artificielle des années 50 jusqu'à aujourd'hui.

Nous aborderons les concepts mathématiques qui sont à la base des modèles d'IA, les problématiques liées à leur développement et les applications dans différents domaines.

L'accent sera principalement porté sur les solutions d'apprentissage profond des années 2010 avec la réémergence des réseaux de neurones, jusqu'aux modèles génératifs les plus récents.

Le passage à l'échelle des modèles d'IA sera traité pour mettre en évidence les ressources nécessaires à leur bon fonctionnement et l'impact social résultant.

Apprentissage automatique d'un schéma numérique pour l'advection d'interfaces.

Matthieu Ancellin (Mews Labs)

Cette présentation est dédiée à un exemple d'intégration de réseau de neurones dans un code de simulation numérique.

Dans le cadre de la simulation d'écoulements à plusieurs fluides non-miscibles, la méthode Volume-Of-Fluid (VOF) représente la distribution des fluides par des fonctions caractéristiques (c'est-à-dire valant 1 lorsqu'un fluide est présent en un point et 0 sinon) qui évoluent dans le temps suivant une équation d'advection. Cependant les schémas numériques Volumes Finis classiques ne sont pas adaptés à la discrétisation de fonctions discontinues comme celles-ci. Des schémas ad-hoc ont donc été proposés, notamment la famille de schémas dits PLIC (Piecewise Linear Interface Capture) qui reconstruit une approximation de la distribution à l'intérieur d'une maille afin de la faire évoluer.

Després et Jourdren [1] ont proposé une approche alternative utilisant un réseau de neurones entrainé sur un grand nombre d'exemples d'interfaces pour calculer le flux numérique entre deux mailles d'un calcul VOF. Cette approche soulève plusieurs questions :

- comment choisir un ensemble représentatif d'interfaces pour l'apprentissage ?
- comment garantir, voire exploiter, les symétries de l'équation d'advection ?
- comment garantir la stabilité de la simulation ?

La présentation se focalisera principalement sur des travaux réalisés sur ce sujet en 2019-2021 [2] sur le problème à deux fluides, ainsi que sur une extension à N fluides [3], mais donnera aussi un aperçu de l'évolution de l'état de l'art depuis lors [4].

[1] Després, B., & Jourdren, H. (2020). Machine learning design of volume of fluid schemes for compressible flows. Journal of Computational Physics, 408, 109275.
[2] Ancellin, M. (2022), Neural networks as volume-of-fluid schemes for multiphase flows, and extension to an arbitrary number of fluids, Présentation au séminaire CEA-GAMNI, Janvier 2022
[3] Ancellin, M., Després, B., & Jaouen, S. (2023). Extension of generic two-component VOF interface advection schemes to an arbitrary number of components. Journal of Computational Physics, 473, 111721.
[4] Pintore, M., & Després, B. (2026). A 3D Machine Learning based Volume Of Fluid scheme without explicit interface reconstruction. Computers & Fluids, 107112

Quantification des incertitudes dans les réseaux de neurones. Application à la prédiction d'inondations de tsunami en France

Pierre Andraud (LRC MESO)

Aujourd'hui, les réseaux de neurones sont devenus un outil incontournable dans la pratique moderne de l'apprentissage automatique. Ce succès tient d'abord à leur grande modularité et à leur performance accrue dans un large domaine de problèmes, notamment en grande dimension. Toutefois, il n'est pas rare que ces modèles "hallucinent" et génèrent des prédictions totalement erronées, en particulier lorsque ceux-ci sont évalués en dehors du support d'apprentissage. En raison de leur formalisme déterministe, impliquant une configuration paramétrique finale θ figée, les réseaux de neurones ne peuvent proposer qu'un candidat F_θ (x) unique pour une entrée x donnée. Cette limitation soulève la question de la fiabilité attribuée à ces modèles. Avec des prédictions dépourvues de degré de confiance, les réseaux de neurones sont dans l'incapacité de détecter les anomalies et de produire en conséquence un avertissement, ce qui limite leur déploiement opérationnel.

La problématique de la fiabilité des modèles d'apprentissage profond fait néanmoins l'objet de recherches depuis plusieurs années. En adaptant les modèles à un cadre stochastique, et en considérant la distribution p(θ|D), qui encode la probabilité d'obtenir une paramétrisation des poids θ sachant que les donnes D ont été observées, il est en effet possible, dans une certaine mesure, de quantifier les incertitudes de prédiction.
Plusieurs adaptations probabilistes des réseaux de neurones et de techniques d'échantillonnage de p(θ|D) ont été proposées dans la littérature pour tenir compte des incertitudes dans les modèles.
L'objectif de cet exposé est d'introduire le public à la mise en œuvre de ces méthodes, en présentant, notamment, deux techniques largement employées dans le domaine : Monte Carlo Dropout et Réseau de neurones bayésien. Une dernière approche, assez récente et prometteuse, MaxWEnt, sera également présentée.

Afin d'expliquer au mieux les notions et concepts clefs dont il sera question pendant cet exposé, l'ensemble de la présentation sera illustré avec une application concrète, à savoir la prédiction rapide d'inondations de tsunami en France.
Cette présentation se veut didactique et accessible à toutes et à tous. Elle sera structurée de la manière suivante :

- Tsunami en France et apprentissage automatique : contexte et enjeux.
- Construction d'un jeu de données de scénarios tsunami en Méditerranée occidentale et implémentation des modèles de prédiction.
- Adaptation des réseaux de neurones à un cadre probabiliste pour la quantification des incertitudes épistémiques. Présentation des trois techniques mentionnées précédemment.
- Application de ces méthodes dans le cadre de notre problème. Comparaison et limite de ces approches.
- Conclusion.

Introduction à l'IA scientifique

Florian De Vuyst (BMBI, UTC et CNRS)

Ce cours est une introduction aux méthodes d'intelligence artificielle et de machine learning pour les problèmes scientifiques.

Il s'adresse avant tout aux débutants qui souhaitent en savoir plus sur les méthodes d'IA et leur implémentation, leur potentiel et leurs limitations.

Le cours est accompagné de travaux pratiques en python : chaque apprenant sera convié à installer un environnement virtuel python permettant l'utilisation de PyTorch sur son laptop (calcul sur CPU ou GPU).

Ce cours ne traitera pas d'AI avancée du type attention/transformers, LLM, RAG (Retrieval augmented Generation) ou agentic AI.

Le contenu du cours est le suivant :

- concepts de base
- régression
- classification binaire et multi-classes
- réseaux de neurones profonds et ResNet
- réduction de dimensionalité
- décomposition tensorielle pour les problèmes paramétrés
- autoencodeurs
- approximation d'opérateurs : DeepONet
- Graph neural networks (GNN)
- découverte d'équations : DMD, xDMD, SINDy, Neural ODE, adjoints
- modèles informés par la physique
- approches structure-preserving
- PINNs
- approches génératives : generative diffusion (DDPM)
- discussion
- environnements logiciels et bibliographie

Des travaux pratiques reprendront les concepts importants vus en cours. Les exercices seront sous forme de Jupyter Notebooks.

L'IA générative pour le développement de codes de simulation : une révolution sous conditions

Thierry Valet (Fondateur et Directeur général, MPhysX OÜ, Tallinn, Estonie, Professeur invité, Université Tohoku (AIMR & CSIS), Sendai)

Depuis deux ans, l'essor des grands modèles de langage (LLMs) a profondément transformé le développement logiciel. La révolution annoncée par les grands laboratoires d'IA est réelle, mais elle ne se concrétise que sous des conditions d'emploi strictes, largement absentes du discours promotionnel. La génération de code n'en est que l'étape terminale : sa qualité est déterminée, tout au long du cycle de développement, par un ensemble de pratiques rigoureuses.

À partir d'un retour d'expérience quotidien et approfondi sur les outils les plus avancés disponibles dans le domaine public, cette session articule quatre de ces pratiques, qui constituent, en combinaison, les conditions nécessaires à des bénéfices réels :

1. L'intégration de l'IA à l'ensemble du cycle de développement, où l'essentiel du travail se déplace vers un dialogue en langage naturel i.e., spécifications fonctionnelles, architecture, plans d'implémentation, automatisation de l'intégration, du test et du déploiement.

2. La segmentation du travail en tâches autonomes, délibérément ajustées au budget de contexte du modèle.

3. La lutte active contre la complaisance du modèle, mode de défaillance réel et prévalent qu'un utilisateur non averti ne perçoit même pas.

4. La vérification systématique de toute production, code comme raisonnement. Le modèle doit être traité comme un générateur de propositions à taux d'erreur non nul.

Illustrée par de nombreux exemples concrets tirés de la pratique de l'intervenant, la présentation défend une thèse simple : à ces conditions, l'ensemble converge vers une productivité décuplée et un code de meilleure qualité que celui issu d'un développement non assisté. Ces conditions, enfin, sont largement invariantes à travers l'ensemble des modèles disponibles, et restent largement valides en relation aux modèles souverains déployés en environnement contraint.

Comprendre et bien utiliser l'IA générative pour le développement de codes scientifiques

Valentina Mejia Gallon (Mews Labs)

Cette présentation est dédiée à la compréhension des grands modèles de langage (LLM) et aux bonnes pratiques pour les utiliser dans le cadre de développement de codes de calcul.

Les assistants de codes fondés sur l'IA générative sont devenus incontournables, mais on les emploie souvent comme des boîtes noires. Pour s'en servir, encore faut-il comprendre ce qu'ils font. La présentation revient donc d'abord sur les fondements de ces modèles : l'évolution récente des modèles de langage, l'architecture Transformer et son mécanisme d'attention [1], et surtout le rôle central du contexte dans la qualité des réponses. L'objectif n'est pas la théorie, mais d'acquérir une intuition suffisante pour saisir les capacités comme les limites de ces outils.

Le cœur de la présentation porte ensuite sur les bonnes pratiques, illustrées à partir d'exemples qui montent en difficulté et donnent davantage de capacités aux LLM. Plusieurs cas d'usage montrent comment un assistant aide à faire la rétroconception d'un code legacy, à générer du code et des tests, à déboguer et à analyser des résultats, et comment cadrer son travail : fournir le bon contexte, faire planifier avant d'écrire, travailler d'abord en lecture seule, fixer des règles de projet, poser des garde-fous et vérifier systématiquement ce qui est produit. On y clarifie aussi une distinction utile : un agent est un modèle équipé d'outils, capable d'agir et de se corriger [3] ; un système d'IA agentique va plus loin en coordonnant plusieurs agents, une mémoire et des outils externes, en s'appuyant sur des techniques telles que la génération augmentée par la récupération (RAG) [2].

En complément, la présentation donnera un aperçu de la façon d'assembler une chaîne entièrement auto-hébergée, reposant sur une extension open source dans l'environnement de développement et une inférence locale de modèles ouverts [4], pour ceux qui souhaitent garder l'assistant et leurs données sur leurs propres machines. Elle s'adresse aux étudiants, chercheurs, ingénieurs et développeurs souhaitant comprendre, évaluer et exploiter l'IA générative dans leurs environnements de simulation numérique.

[1] Vaswani et al. (2017). Attention Is All You Need. NeurIPS. arXiv:1706.03762
[2] Lewis et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS. arXiv:2005.11401
[3] Yao et al. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. ICLR. arXiv:2210.03629
[4] Grattafiori et al. / Meta AI (2024). The Llama 3 Herd of Models. arXiv:2407.21783

Déploiement sécurisé et sur site d'IA génératives, de RAG et d'environnements de développement de codes

David Lugato (CEA CESTA)

Le déploiement d'outils d'intelligence artificielle générative, de Retrieval-Augmented Generation (RAG) et d'environnements de développement de codes en contexte sécurisé et sur site répond à des besoins critiques : souveraineté des données et intégration dans des infrastructures existantes. Ces technologies transforment la productivité, mais leur adoption soulève des défis en matière de sécurité, de latence, et de maintenabilité.

Nous aborderons quelques ingrédients clés pour un déploiement sécurisé en particulier l'utilisation de modèles open-source déployés sur des serveurs locaux pour garantir l'Isolation des données en utilisant des techniques d'optimisation similaires à celles du HPC.
La mise à disposition d'une usine à RAG en environnement sécurisé par une indexation locale des documents sensibles permet de réduire les hallucinations et de tracer les sources.

La méthode Spec-Driven Development (SDD) appuyée sur des IA génératives consiste à définir des spécifications techniques détaillées et structurées (en langage naturel, schémas, ou code formel) qui servent de base à la génération automatique de code, de tests ou de documentation.

L'IA agit comme un assistant expert, capable de proposer des solutions conformes aux exigences, de générer des cas de test, ou même d'identifier des incohérences dans les spécifications. Cette approche renforce la traçabilité, la maintenabilité et la conformité du code, tout en permettant aux développeurs de se concentrer sur la conception architecturale et la validation des besoins métiers. Nous l'avons adaptée tant pour les nouveaux projets que pour les codes de simulation existants.

Physics foundation models: Building cross-domain models for accelerating scientific discovery

Dr. Payel Mukhopadhyay (Department of Applied Mathematics and Theoretical Physics, University of Cambridge, UK)

Most machine-learning models for physics are trained for one equation or one dataset. Move to a different physical system, parameter regime, or source of data, and the model usually has to be trained again from scratch. This lecture asks a simple question: can a model instead learn reusable structure from many physical systems, and then be adapted to a new problem with much less data and training?

I will introduce foundation models for science from a physicist's point of view, without assuming any background in language models. I will discuss the motivation behind Polymathic AI and some of our work on physics foundation models, with examples from fluid dynamics, astronomy, and astrophysics.

A central example in the talk will be Walrus, a spatiotemporal foundation model for physics trained on a broad collection of partial differential equation simulations drawn from several areas of physics, including fluid dynamics and astrophysics. I will use Walrus to explain how such a model is built: how simulation fields are represented, how several past states are used to predict future evolution, and how one pretrained model can be adapted to new physical systems.

I will then show examples from fluid dynamics, including the adaptation of Walrus to Rayleigh-Bénard convection, shear flows, Rayleigh-Taylor instability, and laboratory data. The lecture will be followed by a hands-on tutorial.

Mise en œuvre d'une architecture RAG locale : retour d'expérience

Grégory ROBERT (CEA Gramat)

Dans le domaine de la physique des matériaux, nous utilisons quotidiennement des protocoles de calcul et des modèles physiques complexes qui s'appuient sur des bases de données toujours plus volumineuses et diversifiées. Dans cet environnement, le partage, la recherche et l'exploitation efficace des connaissances constituent un enjeu majeur.

La mise en œuvre d'une intelligence artificielle fondée sur la génération augmentée par récupération (Retrieval-Augmented Generation – RAG) permet d'exploiter efficacement des sources d'information métier au sein d'un environnement local et autonome.
Une architecture RAG repose sur deux composants principaux : un noyau d'intelligence artificielle basé sur un modèle de langage (Large Language Model – LLM) et une base de connaissances métier.

Le LLM est un modèle dont les paramètres ont été appris lors de son entraînement. Il est capable de comprendre des questions formulées en langage naturel, d'analyser les informations qui lui sont fournies et de générer des réponses cohérentes. Cependant, utilisé seul, il ne dispose pas des connaissances spécifiques et reste limité aux connaissances acquises lors de son entraînement.

La base de connaissances constitue la mémoire documentaire du domaine étudié. Elle regroupe les différentes sources d'information métier : documents techniques, bases de données, procédures, nomenclatures, résultats d'études ou données issues de simulations. Afin de permettre une recherche rapide des contenus les plus pertinents, ces informations sont indexées et transformées en représentations vectorielles.

Lorsqu'un utilisateur pose une question, le système RAG recherche d'abord les informations les plus pertinentes dans cette base documentaire. Les éléments identifiés sont ensuite transmis au modèle de langage sous forme de contexte. Le LLM génère alors une réponse en s'appuyant à la fois sur ses capacités générales de compréhension et sur les connaissances spécifiques issues des sources métier.

Cette présentation propose un retour d'expérience sur la mise en œuvre d'une telle architecture en environnement local (calculateur Topaze du TGCC), appliquée à une base de connaissances regroupant différents types de données : sites HTML, nomenclatures, bases de matériaux, études associées à des codes de dynamique rapide et documents PDF.

L'implémentation repose sur un environnement Python utilisant des composants open source dédiés aux architectures RAG, notamment LlamaIndex et llama-cpp-python (licence MIT). Le modèle génératif utilisé est Qwen3-8B-Q4_K_M.gguf, distribué sous licence Apache 2.0. La construction de la base vectorielle repose sur le modèle d'embedding nomic-embed-text-v1.5.f16.gguf, également distribué sous licence Apache 2.0.

Travaux pratiques : prise en main de Claude Code pour le développement d'un code de simulation numérique

Daniel Chauveheid (Mews Labs)

Les assistants de programmation fondés sur l'IA agentique ne se contentent plus de compléter des lignes de code : installés dans le terminal, ils explorent un projet, compilent, exécutent, testent et corrigent, en dialoguant en langage naturel avec le développeur qui garde la décision. Ces travaux pratiques proposent une prise en main guidée de l'un de ces outils, Claude Code (Anthropic), dans le contexte qui est celui des participants : le développement de codes de simulation numérique.
Le support est un mini-solveur d'advection-diffusion 2D dont l'architecture reproduit celle d'un solveur industriel (maillage, champs, conditions aux limites, schémas, boucle en temps, jeu de données, exports), fourni en deux variantes au choix selon les affinités de chacun, C++ ou Fortran 90, et livré volontairement avec une chaîne de compilation cassée et un bug mémoire. Le travail s'effectue en binômes ou trinômes, chaque groupe disposant d'une clé d'accès fournie par l'organisation.

En 1h30, un parcours gradué fait visiter les capacités de l'outil :

- remise en marche autonome du solveur : réparation de la chaîne de compilation, puis diagnostic d'un bug mémoire qui fait planter le cas test ;
- mise en place de la mémoire de projet (CLAUDE.md) et des conventions d'équipe ;
- développement de fonctionnalités physiques validé par un plan avant toute modification : terme source, conditions aux limites périodiques ;
- gestion des entrées/sorties : extension du jeu de données et export VTK visualisé dans ParaView ;
- tests unitaires et vérification numérique : solution manufacturée, erreur L2 et ordre de convergence en maillage ;
- pour finir, au choix : diffusion implicite via le solveur direct creux MUMPS, montée à l'ordre 2 du schéma d'advection, refactoring d'un module hérité, documentation, ou création d'un « skill » encapsulant le workflow de vérification du laboratoire.

Le fil conducteur est l'esprit critique du numéricien : chaque contribution de l'outil est soumise à la compilation sans avertissement, aux tests et à la physique (bilans de masse, ordre de convergence observé confronté à l'ordre théorique). La séance montre ainsi autant comment cadrer un assistant agentique (planifier avant d'écrire, fixer des conventions, vérifier systématiquement) que ce qu'il sait produire.
Aucune expérience préalable des assistants d'IA n'est requise, et aucun fichier source n'est édité à la main : toutes les interventions passent par l'assistant, afin d'en expérimenter au maximum les capacités. Les participants viennent avec leur ordinateur portable préparé à l'avance à l'aide du document de prérequis fourni (environnement Linux ou WSL, compilateurs, ParaView, MUMPS, Claude Code et clé API distribuée par l'organisation). Des pistes « pour aller plus loin » et une correction complète sont remises à l'issue de la séance.

Se rendre à la maison d'hôtes de Cadarache

ADRESSE :

Maison d'hôtes de Cadarache
Route de Vinon/Verdon
13115 Saint Paul lez Durance

VENIR EN VOITURE :

- **Depuis GRENOBLE** :
  - Rejoindre la N75 en direction de SISTERON
  - Sortir et emprunter l'A51 direction AIX-EN-PROVENCE
  - Prendre la sortie 17 CEA CADARACHE

- **Depuis AIX-EN-PROVENCE :**
  - Rejoindre l'A51 en direction de GAP
  - Prendre la sortie 17 CEA CADARACHE

- **Depuis MARSEILLE :**
  - Rejoindre l'A7 en direction d'AIX-EN-PROVENCE puis continuer sur l'A51 direction AIX-EN-PROVENCE/SISTERON
  - Prendre la N296 direction AVIGNON/LYON/SISTERON/GAP
  - Reprendre l'A51 direction GAP
  - Prendre la sortie 17 CEA CADARACHE

- **Depuis la GARE TGV D'AIX EN PROVENCE :**
  - Rejoindre la D9 en direction d'AIX-EN-PROVENCE/LES MILLES
  - Continuer sur l'A51 direction AIX-EN-PROVENCE/SISTERON (puis voir indications **Depuis MARSEILLE**)

